Computing Probabilistic Bisimilarity Distances for Probabilistic Automata

Bacci, Giorgio; Bacci, Giovanni; Larsen, Kim Guldstrand; Mardare, Radu; Tang, Qiyi; van Breugel, Franck

Published in:
30th International Conference on Concurrency Theory (CONCUR 2019)

DOI (link to publication from Publisher):
10.4230/LIPIcs.CONCUR.2019.9

Publication date:
2019

Document Version
Publisher's PDF, also known as Version of record

Computing Probabilistic Bisimilarity Distances for Probabilistic Automata

Giorgio Bacci
Department of Computer Science, Aalborg University, DK
grbacci@cs.aau.dk

Giovanni Bacci
Department of Computer Science, Aalborg University, DK
giovbacci@cs.aau.dk

Kim G. Larsen
Department of Computer Science, Aalborg University, DK
kgl@cs.aau.dk

Radu Mardare
Department of Computer Science, Aalborg University, DK
mardare@cs.aau.dk

Qiyi Tang
Department of Computing, Imperial College, London, UK
qiyi.tang@imperial.ac.uk

Franck van Breugel
DisCoVeri Group, Dept. of Electrical Engineering and Computer Science, York University, Canada
franck@eecs.yorku.ca

Abstract

The probabilistic bisimilarity distance of Deng et al. has been proposed as a robust quantitative generalization of Segala and Lynch's probabilistic bisimilarity for probabilistic automata. In this paper, we present a novel characterization of the bisimilarity distance as the solution of a simple stochastic game. The characterization gives us an algorithm to compute the distances by applying Condon's simple policy iteration on these games. The correctness of Condon's approach, however, relies on the assumption that the games are *stopping*. Our games may be non-stopping in general, yet we are able to prove termination for this extended class of games. Already other algorithms have been proposed in the literature to compute these distances, with complexity in $\mathbf{UP} \cap \mathbf{coUP}$ and **PPAD**. Despite the theoretical relevance, these algorithms are inefficient in practice. To the best of our knowledge, our algorithm is the first practical solution. In the proofs of all the above-mentioned results, an alternative presentation of the Hausdorff distance due to Mémoli plays a central rôle.

2012 ACM Subject Classification Theory of computation → Program verification; Theory of computation → Models of computation; Mathematics of computing → Probability and statistics

Keywords and phrases Probabilistic automata, Behavioural metrics, Simple stochastic games, Simple policy iteration algorithm

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2019.9

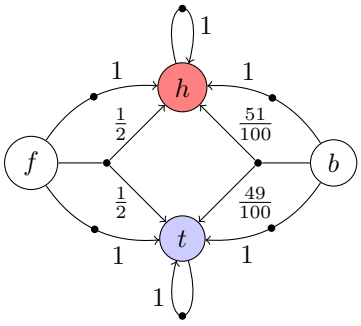
Related Version A full version of the paper is available at `http://arxiv.org/abs/1907.01768`.

Funding Giovanni Bacci and Kim G. Larsen are funded by the ERC-Project LASSO, Radu Mardare is funded by the ASAP Project, Franck van Breugel is supported by Natural Sciences and Engineering Research Council of Canada.

1 Introduction

In [18], Giacalone et al. observed that for reasoning about the behaviour of probabilistic systems, rather than equivalences, a notion of distance is more reasonable in practice since it permits to capture the degree of difference between two states. This observation motivated the study of behavioural pseudometrics, that generalize behavioural equivalences in the sense that, when the distance is zero then the two states are behaviourally equivalent.

The systems we consider in this paper are labelled *probabilistic automata*. This model was introduced by Segala [32] to capture both nondeterminism (hence, concurrency) and probabilistic behaviours. The labels on states are used to express that certain properties of interest hold in particular states. Below we consider an example of probabilistic automaton describing two gamblers, f and b, deciding on which team to bet on in a football match.



Typically the two gamblers know the team to bet on, but occasionally they prefer to toss a coin to make a decision. This is represented by the three probabilistic transitions in the state f. The first two take f to state h (head) or t (tail) with probability one, the last takes f to states h and t with probability $\frac{1}{2}$ each. The difference between f and b is that the former uses a fair coin while the latter uses a biased coin landing on heads with slightly higher probability. Once the decision is taken, it is not changed anymore. This is seen on states h and t which have a single probabilistic transition taking the state to itself with probability one. The states h and t have distinct labels, here represented by colours.

A behavioural pseudometric for probabilistic automata capturing this difference is the *probabilistic bisimilarity distance* by Deng et al. [12], introduced as a robust generalization of Segala and Lynch's probabilistic bisimilarity [33]. The key ingredients of this pseudometric are the Hausdorff metric [19] and the Kantorovich metric [22], respectively used to capture nondeterministic and probabilistic behaviour. In the example above, the behaviours of the states h and t are very different since their labels are different. As a result, their probabilistic bisimilarity distance is one. On the other hand, the behaviours of the states f and b are very similar, which is reflected by the fact that their probabilistic bisimilarity distance is $\frac{1}{100}$.

The first attempt to compute the above distance is due to Chen et al. [9], who proposed a doubly exponential-time procedure to approximate the distances up to any degree of accuracy. The complexity was later improved to **PSPACE** by Chattarjee et al. [6, 7]. Their solutions exploit the decision procedure for the existential fragment of the first-order theory of reals. It is worth noting that [9, 6] consider the pseudometric that does not discount the future (a.k.a. *undiscounted* distance) which entails additional algorithmic challenges. Later, Fu [16] showed that the distances have rational values and that computing the discounted distance can be done in polynomial time by using a value-iteration procedure in combination with the continued fraction algorithm [31, Section 6]. As for the undiscounted distance, he showed that

the threshold problem, i.e., deciding whether the distance is smaller than a given rational, is in $\mathbf{NP} \cap \mathbf{coNP}$[1]. Van Breugel and Worrell [40] have later shown that the problem is in **PPAD**, which is short for polynomial parity argument in a directed graph. Notably, their proof exploits a characterization of the distance as a simple stochastic game. Despite the theoretical relevance of the above algorithms, their implementations are inefficient in practice since they require the use of exact computer arithmetic making it difficult to handle models with more than five states. In this paper, we propose an alternative approach that is inspired by the successful implementations of similar pseudometrics on Markov chains [1, 37, 38].

Our solution is based on a novel characterization of the probabilistic bisimilarity distance as the solution of a simple stochastic game. Stochastic games were introduced by Shapley [34]. A simplified version of these games, called *simple stochastic games*, were studied by Condon [11]. Several algorithms have been proposed to compute the value function of a simple stochastic game, many using policy iteration. Condon [10] proposed an algorithm, known as *simple policy iteration*, that switches only one non-optimal choice per iteration. The correctness of Condon's algorithm, however, relies on the assumption that the game is *stopping*.

It turns out that the simple stochastic games characterizing the probabilistic bisimilarity distances are stopping only when the distances discount the future. In the case the distance is non-discounting, the corresponding games may not be stopping. To recover correctness of the policy iteration procedure we adapt Condon's simple policy iteration algorithm by adding a non-local update of the strategy of the min player and an extra termination condition based on a notion of "self-closed" relation due to Fu [16]. The practical efficiency of our algorithm has been evaluated on a significant set of randomly generated probabilistic automata. The results show that our algorithm performs better than the corresponding iterative algorithms proposed for the discounted distances in [16], even though the theoretical complexity of our proposal is exponential in the worst case (cf. [37]) whereas Fu's is polynomial.

The implementation of the algorithms exploits a coupling structure characterization of the distance that allows us to skip the construction of the simple stochastic game which may result in an exponential blow up of the memory required for storing the game. Finally, we remark that in the proofs of most of the above mentioned results a dual representation of the Hausdorff distance due to Mémoli [26] plays a central rôle.

2 Preliminaries and Notation

The set of functions f from X to Y is denoted by Y^X. We denote by $f[x/y] \in Y^X$ the *update of* f at $x \in X$ with $y \in Y$, defined by $f[x/y](x') = y$ if $x' = x$, otherwise $f[x/y](x') = f(x')$.

A (1-bounded) *pseudometric* on a set X is a function $d\colon X \times X \to [0,1]$ such that, $d(x,x) = 0$, $d(x,y) = d(y,x)$, and $d(x,y) \leq d(x,z) + d(z,y)$ for all $x, y, z \in X$.

Kantorovich lifting. A (discrete) probability distribution on X is a function $\mu\colon X \to [0,1]$ such that $\sum_{x \in X} \mu(x) = 1$, and its support is $supp(\mu) = \{x \in X \mid \mu(x) > 0\}$. We denote by $\mathcal{D}(X)$ the set of probability distributions on X. A pseudometric d on X can be lifted to a pseudometric on probability distributions in $\mathcal{D}(X)$ by means of the Kantorovich lifting [41].

The *Kantorovich lifting* of $d \in [0,1]^{X \times X}$ on distributions $\mu, \nu \in \mathcal{D}(X)$ is defined by

$$\mathcal{K}(d)(\mu,\nu) = \min\left\{ \textstyle\sum_{x,y \in X} d(x,y) \cdot \omega(x,y) \mid \omega \in \Omega(\mu,\nu) \right\}, \qquad \text{(Kantorovich lifting)}$$

[1] The same proof can be adapted to show that the decision problem is in $\mathbf{UP} \cap \mathbf{coUP}$ [17]. Recall that **UP** contains those problems in **NP** with a unique accepting computation.

where $\Omega(\mu,\nu)$ denotes the set of *measure-couplings* for the pair (μ,ν), i.e., distributions $\omega \in \mathcal{D}(X \times X)$ such that, for all $x \in X$, $\sum_{y\in X} \omega(x,y) = \mu(x)$ and $\sum_{y\in X} \omega(y,x) = \nu(x)$. It is a well known fact that the Kantorovich lifting can be equivalently defined by ranging ω over the set of vertices $V(\Omega(\mu,\nu))$ of the polytope $\Omega(\mu,\nu)$. Thus, a minimum is always attained at a vertex. Furthermore, if the set X is finite, the set $V(\Omega(\mu,\nu))$ is finite too.

Hausdorff lifting. A pseudometric d on X can be lifted to nonempty subsets of X by means of the Hausdorff lifting. The *Hausdorff lifting* of $d \in [0,1]^{X\times X}$ on nonempty subsets $A, B \subseteq X$ is defined by

$$\mathcal{H}(d)(A,B) = \max\{\sup_{a\in A} \inf_{b\in B} d(a,b), \sup_{b\in B} \inf_{a\in A} d(a,b)\} \ . \quad \text{(HAUSDORFF LIFTING)}$$

Following Mémoli [26, Lemma 3.1], the Hausdorff lifting has a dual characterization in terms of *set-couplings*[2]. Given $A, B \subseteq X$, a set-coupling for (A,B) is a relation $R \subseteq X \times X$ with left and right projections respectively equal to A and B, i.e., $\{a \mid \exists b \in X.\, a\ R\ b\} = A$ and $\{b \mid \exists a \in X.\, a\ R\ b\} = B$. We write $\mathcal{R}(A,B)$ for the set of the set-couplings for (A,B).

▶ **Theorem 1.** $\mathcal{H}(d)(A,B) = \inf\{\sup_{(a,b)\in R} d(a,b) \mid R \in \mathcal{R}(A,B)\}$.

Clearly, for A, B finite, inf and sup in Theorem 1 can be replaced by min and max, respectively.

3 Probabilistic Automata and Probabilistic Bisimilarity Distance

In this section we recall the definitions of *probabilistic automaton*, Segala and Lynch's *probabilistic bisimulation* [33], and *probabilistic bisimilarity distance* of Deng et al. [12].

▶ **Definition 2.** *A* probabilistic automaton *(PA) is a tuple $\mathcal{A} = (S, L, \to, \ell)$ consisting of a nonempty finite set S of states, a finite set of labels L, a finite total transition relation $\to \subseteq S \times \mathcal{D}(S)$, and a labelling function $\ell\colon S \to L$.*

Note that for simplicity we assume the transition relation $\to$ to be total, that is, for all $s \in S$, there exists a $\mu \in \mathcal{D}(S)$ such that $(s,\mu) \in \to$. For the remainder of this paper we fix a probabilistic automaton $\mathcal{A} = (S, L, \to, \ell)$. We write $s \to \mu$ to denote $(s,\mu) \in \to$ and use $\delta(s)$ to denote the set $\{\mu \mid s \to \mu\}$ of successor distributions of s.

Next we recall the notion of probabilistic bisimilarity due to Segala and Lynch [33] for probabilistic automata. Their definition exploits the notion of lifting of a relation $R \subseteq S \times S$ on states to a relation $\tilde{R} \subseteq \mathcal{D}(S) \times \mathcal{D}(S)$ on probability distributions on states, originally introduced by Jonsson and Larsen [20], and defined by $\mu\ \tilde{R}\ \nu$ if there exists a measure-coupling $\omega \in \Omega(\mu,\nu)$ such that $supp(\omega) \subseteq R$.

▶ **Definition 3.** *A relation $R \subseteq S \times S$ is a* probabilistic bisimulation *if whenever $s\ R\ t$,*

- *$\ell(s) = \ell(t)$,*
- *if $s \to \mu$ then there exists $t \to \nu$ such that $\mu\ \tilde{R}\ \nu$, and*
- *if $t \to \nu$ then there exists $s \to \mu$ such that $\mu\ \tilde{R}\ \nu$.*

Two states $s, t \in S$ are probabilistic bisimilar*, written $s \sim t$, if they are related by some probabilistic bisimulation.*

[2] Mémoli uses the terminology "correspondence." To avoid confusion, we adopted the same terminology used in [29, Section 10.6].

Intuitively, two states are probabilistic bisimilar if they have the same label and each transition of the one state to a distribution μ can be matched by a transition of the other state to a distribution ν assigning the same probability to states that behave the same, and vice versa. Probabilistic bisimilarity is an equivalence relation and the largest probabilistic bisimulation.

Deng et al. [12] proposed a family of 1-bounded pseudometrics $\mathbf{d}_\lambda$, parametric on a *discount factor* $\lambda \in (0,1]$, called *probabilistic bisimilarity pseudometrics*. The pseudometrics $\mathbf{d}_\lambda$ are defined as the least fixed-point of the functions $\Delta_\lambda \colon [0,1]^{S\times S} \to [0,1]^{S\times S}$

$$\Delta_\lambda(d)(s,t) = \begin{cases} 1 & \text{if } \ell(s) \neq \ell(t) \\ \lambda \cdot \mathcal{H}(\mathcal{K}(d))(\delta(s),\delta(t)) & \text{otherwise}\,. \end{cases}$$

The well-definition of $\mathbf{d}_\lambda$ follows by Knaster-Tarski's fixed point theorem, given the fact that Δ_λ is a monotone function on the complete partial order of $[0,1]$-valued functions on $S \times S$ ordered point-wise by $d \sqsubseteq d'$ iff for all $s,t \in S$, $d(s,t) \leq d'(s,t)$.

The fact that probabilistic bisimilarity distances provide a quantitative generalization of bisimilarity is captured by the following theorem due to Deng et al. [12, Corollary 2.14].

▶ **Theorem 4.** *For all $\lambda \in (0,1]$, $\mathbf{d}_\lambda(s,t) = 0$ if and only if $s \sim t$.*

4 Probabilistic Bisimilarity Distance as a Simple Stochastic Game

A *simple stochastic game* (SSG) consists of a finite directed graph whose vertices are partitioned into sets of *0-sinks*, *1-sinks*, *max vertices*, *min vertices*, and *random vertices*. The game is played by two players, the *max player* and the *min player*, with a single token. At each step of the game, the token is moved from a vertex to one of its successors. At a min vertex the min player chooses the successor, at a max vertex the max player chooses the successor, and at a random vertex the successor is chosen randomly according to a prescribed probability distribution. The max player wins a play of the game if the token reaches a 1-sink and the min player wins if the play reaches a 0-sink or continues forever without reaching a sink. Since the game is stochastic, the max player tries to maximize the probability of reaching a 1-sink whereas the min player tries to minimize that probability.

▶ **Definition 5.** *A* simple stochastic game *is a tuple (V,E,P) consisting of*

- *a finite directed graph (V,E) such that*
  - *V is partitioned into the sets: V_0 of* 0-sinks*, V_1 of* 1-sinks*, $V_{\max}$ of* max vertices*, $V_{\min}$ of* min vertices*, and V_{rnd} of* random vertices*;*
  - *the vertices in V_0 and V_1 have outdegree zero and all other vertices have outdegree at least one, and*
- *a function $P \colon V_{\mathrm{rnd}} \to \mathcal{D}(V)$ such that for all $v \in V_{\mathrm{rnd}}$ and $w \in V$, $P(v)(w) > 0$ iff $(v,w) \in E$.*

A *strategy*, also known as *policy*, for the min player is a function $\sigma_{\min} \colon V_{\min} \to V$ that assigns the target of an outgoing edge to each min vertex, that is, for all $v \in V_{\min}$, $(v, \sigma_{\min}(v)) \in E$. Likewise, a strategy for the max player is a function $\sigma_{\max} \colon V_{\max} \to V$ that assigns the target of an outgoing edge to each max vertex. These strategies are known as *pure stationary* strategies. We can restrict ourselves to these strategies since both players of a simple stochastic game have optimal strategies of this type (see, for example, [25]).

Such strategies determine a sub-game in which each max vertex and each min vertex has outdegree one (see [11, Section 2] for details). Such a game can naturally be viewed as a Markov chain. We write $\phi_{\sigma_{\min},\sigma_{\max}} \colon V \to [0,1]$ for the function that gives the probability of a vertex in this Markov chain to reach a 1-sink.

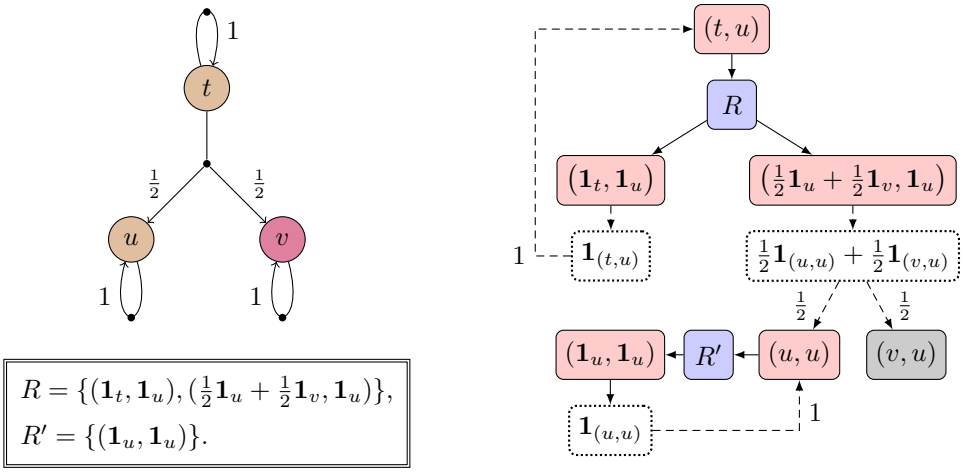


Figure 1 (Top left:) A probabilistic automaton and (Right:) the associated simple stochastic game constructed as in Definition 8 for $\lambda = 1$ (only the portion reachable from (t,u) is shown), where $\mathbf{1}_x$ denotes the Dirac distribution concentrated at x.

The *value function* $\phi\colon V \to [0,1]$ of a SSG is defined as $\min_{\sigma_{\min}} \max_{\sigma_{\max}} \phi_{\sigma_{\min},\sigma_{\max}}$. It is folklore that the value function of a simple stochastic game can be characterised as the least fixed point of the following monotone function (see, for example, [21, Section 2.2 and 2.3]).

▶ **Definition 6.** *The function* $\Phi\colon [0,1]^V \to [0,1]^V$ *is defined by*

$$\Phi(f)(v) = \begin{cases} 0 & \text{if } v \in V_0 \\ 1 & \text{if } v \in V_1 \\ \max_{(v,w)\in E} f(w) & \text{if } v \in V_{\max} \\ \min_{(v,w)\in E} f(w) & \text{if } v \in V_{\min} \\ \sum_{(v,w)\in E} P(v)(w)\, f(w) & \text{if } v \in V_{\mathrm{rnd}} \end{cases}$$

▶ **Proposition 7.** *The function* Φ *is monotone and nonexpansive.*

A Probabilistic Bisimilarity Game. Fix a probabilistic automaton $\mathcal{A}$ and $\lambda \in (0,1]$. We will characterise the probabilistic bisimilarity distance as values of a simple stochastic game, which we call the *probabilistic bisimilarity game*, where the min player tries to show that two states are probabilistic bisimilar, while the max player tries to prove the opposite.

In our probabilistic bisimilarity game, there is a vertex (s,t) for each pair states s and t in $\mathcal{A}$. If $\ell(s) \neq \ell(t)$ then the vertex (s,t) is a 1-sink. Otherwise, (s,t) is a min vertex. In this vertex, the min player selects a set $R \in \mathcal{R}(\delta(s), \delta(t))$ of pairs of transitions. This set R captures potential matchings of transitions from state s and state t. Subsequently, the max player chooses a pair of transitions from the set R. Once the max player has chosen a pair (μ,ν) from the set R corresponding to the transitions $s \to \mu$ and $t \to \nu$, the min player can choose a measure-coupling $\omega \in \Omega(\mu,\nu)$. To ensure that the game graph is finite, we restrict our attention to the vertices $V(\Omega(\mu,\nu))$ of the polytope $\Omega(\mu,\nu)$. Such a measure-coupling ω captures a matching of the probability distributions μ and ν. Recall that a coupling is a probability distribution on $S \times S$. From a random vertex ω, the game proceeds to vertex (u,v) with probability $\lambda \cdot \omega(u,v)$ and to the 0-sink vertex $\bot$ with probability $1-\lambda$. Intuitively,

the choices of $R \in \mathcal{R}(\delta(s), \delta(t))$ and then $(\mu, \nu) \in R$, performed respectively by the min and the max player, correspond to the min and max of Theorem 1; analogously, the selection of $\omega \in V(\Omega(\mu, \nu))$ by the min player models the min in the definition of the Kantorovich lifting.

Formally, our probabilistic bisimilarity game for the automaton $\mathcal{A}$ is defined as follows.

▶ **Definition 8.** *Let* $\lambda \in (0, 1]$. *The* probabilistic bisimilarity game (V, E, P) *is defined by*

- $V_0 = \{\bot\}$,
- $V_1 = \big\{(s,t) \in S \times S \mid \ell(s) \neq \ell(t)\big\}$,
- $V_{\max} = \bigcup \big\{\mathcal{R}(\delta(s), \delta(t)) \mid (s,t) \in V_{\min}\big\}$,
- $V_{\min} = \big\{(s,t) \in S \times S \mid \ell(s) = \ell(t)\big\} \cup \bigcup \big\{R \mid R \in V_{\max}\big\}$,
- $V_{\mathrm{rnd}} = \bigcup \big\{V(\Omega(\mu, \nu)) \mid (\mu, \nu) \in V_{\min}\big\}$,

$$
\begin{aligned}
E = {} & \big\{((s,t), R) \mid (s,t) \in V_{\min} \wedge R \in \mathcal{R}(\delta(s), \delta(t))\big\} \cup \\
& \big\{(R, (\mu, \nu)) \mid R \in V_{\max} \wedge (\mu, \nu) \in R\big\} \cup \\
& \big\{((\mu, \nu), \omega) \mid (\mu, \nu) \in V_{\min} \wedge \omega \in V(\Omega(\mu, \nu))\big\} \cup \\
& \big\{(\omega, (u, v)) \mid \omega \in V_{\mathrm{rnd}} \wedge (u, v) \in \mathit{supp}(\omega)\big\} \cup \big\{(\omega, \bot) \mid \omega \in V_{\mathrm{rnd}}\big\},
\end{aligned}
$$

and, for all $\omega \in V_{\mathrm{rnd}}$ *and* $(s,t) \in \mathit{supp}(\omega)$, $P(\omega)((s,t)) = \lambda \cdot \omega(s,t)$ *and* $P(\omega)(\bot) = 1 - \lambda$.

By construction of the probabilistic bisimilarity game, there is a direct correspondence between the function Φ from Definition 6 associated to the probabilistic bisimilarity game and the function Δ_λ from Section 3 associated to the probabilistic automaton. From this correspondence it is straightforward that the respective least fixed points of Φ and Δ_λ agree, that is, the probabilistic bisimilarity distances of a probabilistic automaton are the values of the corresponding vertices of the probabilistic bisimilarity game.

▶ **Theorem 9.** *For all* $s, t \in S$, $\mathbf{d}_\lambda(s,t) = \phi(s,t)$.

The proof is similar to that of [40, Theorem 14].

Consider a state pair (s,t) with $s \sim t$. By Theorem 4, $\mathbf{d}_\lambda(s,t) = 0$. Hence, from Theorem 9 we can conclude that $\phi(s,t) = 0$. Therefore, by pre-computing probabilistic bisimilarity, (s,t) can be represented as a 0-sink, rather than a min vertex. For example, in Figure 1 this amounts to turn (u,u) into a 0-sink and disconnect it from its successors.

Games similar to the above introduced probabilistic bisimilarity game have been presented in [14, 40, 15, 24]. The game presented by van Breugel and Worrell in [40] is most closely related to our game. They also consider probabilistic automata and map a probabilistic automaton to a simple stochastic game. The only difference is that they use the original definition of the Hausdorff distance, whereas we use Mémoli's alternative characterization. The games described in [14, 15, 24] are not stochastic. Desharnais, Laviolette and Tracol [14] define an ϵ-probabilistic bisimulation game for probabilistic automata, where $\epsilon > 0$ captures the maximal amount of difference in behaviour that is allowed. Their measure of difference in behaviour is incomparable to our probabilistic bisimilarity distances (see [14, Section 6]). König and Mika-Michalski [24] generalize the game of Desharnais et al. in a categorical setting so that it is applicable to a large class of systems including probabilistic automata. Fijalkow, Klin and Panangaden [15] consider a more restricted class of systems, namely systems with probabilities but without nondeterminism. In the games in [14, 15, 24] players choose sets of states, a phenomenon that one does not encounter in our game.

5 A Coupling Characterisation of the Bisimilarity Distance

In this section we provide an alternative characterisation for the probabilistic bisimilarity distance $\mathbf{d}_\lambda$ based on the notion of coupling structure for a probabilistic automaton. This characterisation generalises the one by Chen et al. [8, Theorem 8] (see also [1, Theorem 8]) for the bisimilarity pseudometric of Desharnais et al. [13] for Markov chains. Our construction exploits Mémoli's dual characterisation of the Hausdorff distance (Theorem 1).

▶ **Definition 10.** *A* coupling structure *for $\mathcal{A}$ is a tuple $\mathcal{C} = (f, \rho)$ consisting of*
- *a map $f\colon \mathcal{D}(S) \times \mathcal{D}(S) \to \mathcal{D}(S \times S)$ such that, for all $\mu, \nu \in \mathcal{D}(S)$, $f(\mu,\nu) \in \Omega(\mu,\nu)$, and*
- *a map $\rho\colon S \times S \to 2^{\mathcal{D}(S)\times\mathcal{D}(S)}$, such that for all $s,t \in S$, $\rho(s,t) \in \mathcal{R}(\delta(s),\delta(t))$.*

For convenience, the components f and ρ of a coupling structure will be respectively called *measure-coupling map* and *set-coupling map*.

A coupling structure $\mathcal{C} = (f,\rho)$ induces a probabilistic automaton $\mathcal{A}_\mathcal{C}$ on $S \times S$ with transition relation $\to_\mathcal{C} \subseteq (S\times S) \times \mathcal{D}(S\times S)$, defined as $(s,t) \to_\mathcal{C} f(\mu,\nu)$ if $(\mu,\nu) \in \rho(s,t)$.

Let $\lambda \in (0,1]$. For each $\mathcal{C}$ we define the function $\Gamma^\mathcal{C}_\lambda\colon [0,1]^{S\times S} \to [0,1]^{S\times S}$ as

$$\Gamma^\mathcal{C}_\lambda(d)(s,t) = \begin{cases} 1 & \text{if } \ell(s) \neq \ell(t) \\ \lambda \cdot \max\{\sum_{u,v\in S} d(u,v)\cdot\omega(u,v) \mid (s,t) \to_\mathcal{C} \omega\} & \text{otherwise.} \end{cases}$$

One can easily verify that $\Gamma^\mathcal{C}_\lambda$ is well-defined and monotonic. Thus, by Knaster-Tarski's fixed point theorem, $\Gamma^\mathcal{C}_\lambda$ has a least fixed point, denoted by $\gamma^\mathcal{C}_\lambda$. As in [1], we call $\gamma^\mathcal{C}_\lambda$ the *λ-discounted discrepancy* w.r.t. $\mathcal{C}$ or simply *λ-discrepancy*.

▶ Remark 11. Note that, $\gamma^\mathcal{C}_1(s,t)$ is the maximal probability of reaching a pair of states (u,v) in the probabilistic automaton $\mathcal{A}_\mathcal{C}$ such that $\ell(u) \neq \ell(v)$ by starting from the state pair (s,t). It is well known that the maximal reachability probability can be computed in polynomial-time as the optimal solution of a linear program (see [5, Chapter 10] or [30, Chapter 6]). The linear program can be trivially generalized to compute $\gamma^\mathcal{C}_\lambda$, for any $\lambda \in (0,1]$.

▶ **Lemma 12.** *Let $\lambda \in (0,1]$. Then, $\Delta_\lambda(\gamma^\mathcal{C}_\lambda) \sqsubseteq \gamma^\mathcal{C}_\lambda$ for all coupling structures $\mathcal{C}$ of $\mathcal{A}$.*

The next lemma shows that the probabilistic bisimilarity distance can be characterised as the λ-discrepancy for a *vertex coupling structure*, that is, a coupling structure $\mathcal{C} = (f,\rho)$ such that $f(\mu,\nu) \in V(\Omega(\mu,\nu))$ for all $\mu,\nu \in \mathcal{D}(S)$.

▶ **Lemma 13.** *Let $\lambda \in (0,1]$. Then, $\mathbf{d}_\lambda = \gamma^\mathcal{C}_\lambda$ for some vertex coupling structure $\mathcal{C}$.*

▶ **Theorem 14.** *Let $\lambda \in (0,1]$. Then, the following hold:*
1. $\mathbf{d}_\lambda = \sqcap\{\gamma^\mathcal{C}_\lambda \mid \mathcal{C}$ *coupling structure for* $\mathcal{A}\}$*;*
2. *$s \sim t$ iff $\gamma^\mathcal{C}_\lambda(s,t) = 0$ for some vertex coupling structure $\mathcal{C}$ for $\mathcal{A}$.*

Proof. (1) follows by Knaster-Tarski's fixed point theorem, Lemmas 12, and 13; (2) follows by Theorem 4 and Lemma 13. ◀

Note that together with Lemma 13, Theorem 14.1 states that $\mathbf{d}_\lambda$ can obtained as the minimal λ-discrepancy obtained by ranging over the subset of vertex coupling structures.

▶ Remark 15 (On the relation with probabilistic bisimilarity games). The coupling structure characterization of the distance is strongly related to the simple stochastic game characterization presented in Section 4. Indeed, the notion of vertex coupling structure captures essentially the strategies for the min player on a probabilistic bisimilarity game in the following sense: the measure-coupling map component describes the strategy on the vertices of the form $(\mu,\nu) \in R$ for some $R \in V_{\max}$, while the set-coupling map deals with the description of the strategy on the min vertices $(s,t) \in S \times S$. The discrepancy $\gamma^\mathcal{C}_1$ captures the value w.r.t an optimal strategy for the max player when the min player has fixed their strategy a priori.

6 Computing the Bisimilarity Distance

We describe a procedure for computing the bisimilarity distances based on Condon's simple policy iteration algorithm [10]. Our procedure extends a similar one proposed in [37, 1] for computing the bisimilarity distance of Desharnais et al. [13] for Markov chains. The extension takes into account the additional presence of nondeterminism in the choice of the transitions.

Condon's simple policy iteration algorithm computes the values of a simple stochastic game provided that the game is *stopping*, i.e., for each pair of strategies for the min and max players the token reaches a 0-sink or 1-sink vertex with probability one.

As we have shown in Theorem 9, the probabilistic bisimilarity distances are the values of the corresponding vertices in the simple stochastic game given in Definition 8. Thus, if we prove that the game is stopping we can apply Condon's simple policy iteration algorithm to compute the probabilistic bisimilarity distances.

▶ **Proposition 16.** *For $\lambda \in (0,1)$, the simple stochastic game in Definition 8 is stopping.*

However, for $\lambda = 1$ the game in Definition 8 may not be stopping as shown below.

▶ **Example 17.** Consider the probabilistic automaton in Figure 1 and its associated probabilistic bisimilarity game. By choosing a strategy $\sigma_{\max}$ for the max player such that $\sigma_{\max}(R) = (\mathbf{1}_t, \mathbf{1}_u)$, the vertex (t,u) has probability zero to reach a sink. This can be seen in Figure 1, since there are no paths using the edge $(R, (\mathbf{1}_t, \mathbf{1}_u))$ leading to a sink.

In [37], by imposing the bisimilar state pairs to be 0-sinks, for the case of Markov chains the simple stochastic game was proven to be stopping. This method does not generalize to probabilistic automata. Indeed, Example 17 provides a counterexample even when bisimilar state pairs are 0-sinks.

In the remainder of the section, we provide a general algorithm to compute the bisimilarity distance for every $\lambda \in (0,1]$, by adapting Condon's simple policy iteration algorithm. Our solution will exploit the coupling characterization of the distance discussed in Section 5. This allows us to skip the construction of the simple stochastic game which may have size exponential in the number of states of the automaton.

6.1 Simple Policy Iteration Strategy

Condon's algorithm iteratively updates the strategies of the min and max players in turn, on the basis of the current over-approximation of the value of the game. Next we show how Condon's policy updates can be performed directly on coupling structures.

For the update of the coupling structure, will use measure-coupling maps of the form $k(d)(\mu,\nu) \in V(\Omega(\mu,\nu))$ and a set-coupling $h(d)(s,t) \in \mathcal{R}(\delta(s),\delta(t))$ such that

$$k(d)(\mu,\nu) \in \operatorname{argmin}\left\{ \textstyle\sum_{u,v\in S} \omega(u,v)\cdot d(u,v) \mid \omega \in V(\Omega(\mu,\nu)) \right\}, \text{ and} \tag{1}$$

$$h(d)(s,t) \in \operatorname{argmin}\left\{ \max_{(\mu,\nu)\in R} \mathcal{K}(d)(\mu,\nu) \mid R \in \mathcal{R}(\delta(s),\delta(t)) \right\}. \tag{2}$$

for $d\colon S\times S \to [0,1]$, $\mu,\nu \in \mathcal{D}(S)$, and $s,t \in S$.

The following lemma explains how the above ingredients can be used by the min player to improve their strategy.

▶ **Lemma 18.** *Let $\mathcal{C} = (f,\rho)$. If there exist $s,t \in S$ such that $\Delta_\lambda(\gamma^{\mathcal{C}}_\lambda)(s,t) < \gamma^{\mathcal{C}}_\lambda(s,t)$ then, $\gamma^{\mathcal{D}}_\lambda \sqsubset \gamma^{\mathcal{C}}_\lambda$ for a coupling structure $\mathcal{D} = (k(\gamma^{\mathcal{C}}_\lambda), \rho[(s,t)/R])$, where $R = h(\gamma^{\mathcal{C}}_\lambda)(s,t)$.*

Algorithm 1: Simple policy iteration algorithm computing $\mathbf{d}_\lambda$ for $\lambda \in (0,1)$.

1 Initialise $\mathcal{C} = (f, \rho)$ as an arbitrary vertex coupling structure for $\mathcal{A}$
2 **while** $\exists (s,t).\, \Delta_\lambda(\gamma_\lambda^{\mathcal{C}})(s,t) < \gamma_\lambda^{\mathcal{C}}(s,t)$ **do**
3 $\quad R \leftarrow h(\gamma_\lambda^{\mathcal{C}})(s,t)$
4 $\quad \mathcal{C} \leftarrow \big(k(\gamma_\lambda^{\mathcal{C}}),\, \rho[(s,t)/R]\big)$ `/* update coupling structure */`
5 **end**
6 **return** $\gamma_\lambda^{\mathcal{C}}$ `/*` $\gamma_\lambda^{\mathcal{C}} = \mathbf{d}_\lambda$ `*/`

Lemma 18 suggests that $\mathcal{C} = (f, \rho)$ can be improved by replacing the measure-coupling map f with $k(\gamma_\lambda^{\mathcal{C}})$ and updating the set-coupling map ρ at (s,t) with $R = h(\gamma_\lambda^{\mathcal{C}})(s,t)$.

Note that a measure-coupling $k(d)(\mu,\nu)$ satisfying (1) can be computed by solving a linear program and ensuring that the optimal solution is a vertex of the polytope $\Omega(\mu,\nu)$ [28, 23]. A set-coupling $h(d)(s,t)$ satisfying (2) is the following:

$$R = \{(\mu, \phi(\mu)) \mid \mu \in \delta(s)\} \cup \{(\psi(\nu), \nu) \mid \nu \in \delta(t)\} \in \mathcal{R}(\delta(s), \delta(t))\,, \tag{3}$$

where ϕ, ψ are such that $\phi(\mu) \in \operatorname{argmin}_{\nu \in \delta(t)} \mathcal{K}(d)(\mu,\nu)$ and $\psi(\nu) \in \operatorname{argmin}_{\mu \in \delta(s)} \mathcal{K}(d)(\mu,\nu)$. The following lemma justifies our choice of $h(d)(s,t)$.

▶ **Lemma 19.** *Let R be as in* (3)*. Then $\mathcal{H}(\mathcal{K}(d))(\delta(s),\delta(t)) = \max_{(\mu,\nu)\in R} \mathcal{K}(d)(\mu,\nu)$.*

▶ Remark 20. The update procedure entailed by Lemma 18 can be performed in polynomial-time in the size of the probabilistic automaton $\mathcal{A}$. Indeed, $k(d)(\mu,\nu)$ can be obtained by solving a transportation problem in polynomial time [28, 23]. As for $h(d)(s,t)$, one can obtain $\phi(\mu)$ (resp. $\psi(\nu)$) by computing $\mathcal{K}(d)(\mu,\nu)$ in polynomial time and selecting the ν (resp. μ) ranging over $\delta(t)$ (resp. $\delta(s)$) that achieves the minimum.

Discounted case. The simple policy iteration algorithm for computing $\mathbf{d}_\lambda$ in the case $\lambda < 1$ is presented in Algorithm 1. The procedure starts by computing an initial vertex coupling structure $\mathcal{C}_0$ (line 1), e.g., by using the North-West corner method in polynomial time (see [35, pg. 180]). Then it continues by iteratively generating a sequence $\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_n$ of vertex coupling structures where $\mathbf{d}_\lambda = \gamma_\lambda^{\mathcal{C}_n}$. At each iteration, the current coupling structure $\mathcal{C}_i$ is tested for optimality (line 2) by checking whether the corresponding λ-discrepancy $\gamma_\lambda^{\mathcal{C}_i}$ is a fixed point for Δ_λ. If there exists $(s,t) \in S$ violating the equality $\gamma_\lambda^{\mathcal{C}_i} = \Delta_\lambda(\gamma_\lambda^{\mathcal{C}_i})$, it constructs $\mathcal{C}_{i+1}$ by updating $\mathcal{C}_i$ at (s,t) as prescribed by Lemma 18 (line 4). This guarantees that $\gamma_\lambda^{\mathcal{C}_i} \sqsupset \gamma_\lambda^{\mathcal{C}_{i+1}}$, i.e., a strict improvement of the λ-discrepancy towards the minimal one.

Termination follows by the fact that there are only finitely many vertex coupling structures for $\mathcal{A}$. Furthermore, the correctness of the output of the algorithm is due to the fact that, Δ_λ has a unique fixed point when $0 \le \lambda < 1$.

▶ **Theorem 21.** *Let $\lambda \in (0,1)$. Algorithm 1 is terminating and computes $\mathbf{d}_\lambda$.*

Undiscounted case. For $\lambda = 1$, the termination condition of the simple policy-iteration algorithm of Section 6.1 is not sufficient to guarantee correctness, since Algorithm 1 may terminate prematurely by returning a fixed point of Δ_1 that is not the minimal one.

Towards a way to obtain a stronger termination condition, we introduce the notion of self-closed relations w.r.t. a fixed point for Δ_1, originally due to [16].

▶ **Definition 22.** *A relation $M \subseteq S \times S$ is* self-closed w.r.t. $d = \Delta_1(d)$ *if, whenever $s\ M\ t$,*

- $\ell(s) = \ell(t)$ *and* $d(s,t) > 0$,
- *if $s \to \mu$ and $d(s,t) = \min_{\nu' \in \delta(t)} \mathcal{K}(d)(\mu,\nu')$ then there exists $t \to \nu$ and $\omega \in \Omega(\mu,\nu)$ such that $d(s,t) = \sum_{u,v \in S} d(u,v) \cdot \omega(u,v)$ and $supp(\omega) \subseteq M$,*
- *if $t \to \nu$ and $d(s,t) = \min_{\mu' \in \delta(s)} \mathcal{K}(d)(\mu',\nu)$ then there exists $s \to \mu$ and $\omega \in \Omega(\mu,\nu)$ such that $d(s,t) = \sum_{u,v \in S} d(u,v) \cdot \omega(u,v)$ and $supp(\omega) \subseteq M$.*

Two states are self-closed w.r.t d, written $s \approx_d t$, if they are related by some self-closed relation w.r.t. d.

It can be easily shown that $\approx_d$ is the largest self-closed relation w.r.t. d. Note that the concept of self-closeness above is defined only for fixed points of Δ_1. As remarked in [16], the largest self-closed relation $\approx_d$ can be computed in polynomial time by using partition refinement techniques similar to those employed to compute the largest bisimilarity relation.

The next lemma states that if for a fixed point $d = \Delta_1(d)$ the relation $\approx_d$ is nonempty, then d is not the least fixed point of Δ_1.

▶ **Lemma 23.** *Let $d = \Delta_1(d)$. If there exists a nonempty self-closed relation M w.r.t. d, then there exists $d_M \sqsubset d$ such that $\Delta_1(d_M) \sqsubseteq d_M$. Moreover, d_M can be computed in polynomial time in the size of the probabilistic automaton $\mathcal{A}$.*

The proof of Lemma 23 is essentially that of [16, Theorem 3]. Given a nonempty self-closed relation M w.r.t. d, the above result can be used to obtain a prefix point of Δ_1, namely d_M, that improves d towards the search of the least fixed point. The prefix point d_M of Lemma 23 is obtained from d by subtracting a constant $\theta > 0$ from all the distances computed at pairs of states in M:

$$d_M(s,t) = \begin{cases} d(s,t) - \theta & \text{if } (s,t) \in M\,, \\ d(s,t) & \text{if } (s,t) \notin M\,, \end{cases}$$

where θ is the maximal value satisfying the following set of inequalities

$$\begin{aligned} &\theta \leq d(s,t) - \min_{\nu' \in \delta(t)} \mathcal{K}(d)(\mu,\nu') && \text{for all } (s,t) \in M \text{ and } \mu \in \delta(s), \\ &\theta \leq d(s,t) - \min_{\mu' \in \delta(s)} \mathcal{K}(d)(\mu',\nu) && \text{for all } (s,t) \in M \text{ and } \nu \in \delta(t), \\ &\theta \leq d(s,t) && \text{for all } (s,t) \in M\,. \end{aligned}$$

The fact that d_M is a prefix point follows since M is a self-closed relation.

The following lemma provides us with a termination condition for the simple policy iteration algorithm to compute $\mathbf{d}_1$. Indeed, according to it, if d is a fixed point of Δ_1, we can assert that d is equal to bisimilarity distance $\mathbf{d}_1$ by simply checking that the maximal self-closed relation w.r.t. d is empty.

▶ **Lemma 24.** *If $d = \Delta_1(d)$ and $\approx_d = \emptyset$, then $d = \mathbf{d}_1$.*

Algorithm 2 extends the procedure described in Section 6.1 by encapsulating the policy iteration update (lines 4–7) into an outer-loop (lines 3–15) that is responsible to check whether the fixed point $\gamma_1^{\mathcal{C}_i}$ returned is the minimal one. According to Lemma 12, $\Delta_1(\gamma_1^{\mathcal{C}_i}) \sqsubseteq \gamma_1^{\mathcal{C}_i}$. Hence, when we reach line 8, we have that $\Delta_1(\gamma_1^{\mathcal{C}_i}) = \gamma_1^{\mathcal{C}_i}$. Therefore, by Lemmas 23 and 24, $\gamma_1^{\mathcal{C}_i} = \mathbf{d}_1$ if and only if $M = \approx_{\gamma_1^{\mathcal{C}_i}}$ is empty. If M is empty, we set the variable isMin to *true* (line 10) causing the outer-loop to terminate. Otherwise, we construct $d = (\gamma_1^{\mathcal{C}_i})_M$ as in

Algorithm 2: Simple policy iteration algorithm computing $\mathbf{d}_1$.

1 Initialise $\mathcal{C} = (f, \rho)$ as an arbitrary vertex coupling structure for $\mathcal{A}$
2 isMin $\leftarrow$ *false*
3 **while** $\neg$isMin **do**
4 **while** $\exists (s,t).\, \Delta_1(\gamma_1^{\mathcal{C}})(s,t) < \gamma_1^{\mathcal{C}}(s,t)$ **do**
5 $R \leftarrow h(\gamma_1^{\mathcal{C}})(s,t)$
6 $\mathcal{C} \leftarrow \big(k(\gamma_1^{\mathcal{C}}),\, \rho[(s,t)/R]\big)$ `/* update coupling structure */`
7 **end**
8 Let $M \leftarrow \approx_{\gamma_1^{\mathcal{C}}}$ `/* note that` $\gamma_1^{\mathcal{C}} = \Delta_1(\gamma_1^{\mathcal{C}})$ `*/`
9 **if** $M = \emptyset$ **then**
10 isMin $\leftarrow$ *true* `/*` $\gamma_1^{\mathcal{C}} = \mathbf{d}_1$ `*/`
11 **else**
12 Compute $d = (\gamma_1^{\mathcal{C}})_M$ as in Lemma 23
13 Re-initialise $\mathcal{C}$ as a vertex coupling structure s.t. $\Gamma_1^{\mathcal{C}}(d) = \Delta_1(d)$
14 **end**
15 **end**
16 **return** $\gamma_1^{\mathcal{C}}$

Lemma 23 (line 12) and re-start the inner-loop from a vertex coupling structure $\mathcal{C}_{i+1}$ such that $\Gamma_1^{\mathcal{C}_{i+1}}(d) = \Delta_1(d)$ (line 13) (e.g., by using $\mathcal{C}_{i+1} = (k(d), \rho)$ where $\rho(s,t) = h(d)(s,t)$ for all $s, t \in S$). As proven in Theorem 25, $\gamma_1^{\mathcal{C}_i} \sqsupset \gamma_1^{\mathcal{C}_{i+1}}$. This guarantees a strict improvement of the discrepancy towards the minimal one. Termination of Algorithm 2 is justified by similar arguments as for the discounted case.

▶ **Theorem 25.** *Algorithm 2 is terminating and computes* $\mathbf{d}_1$.

6.2 Experimental Results

In this section, we evaluate the performance of the simple policy iteration algorithms on a collection of randomly generated probabilistic automata. All the algorithms have been implemented in Java and the source code is publicly available[3].

The performance of Algorithm 1 has been compared with an implementation of the *value iteration algorithm* proposed by Fu [16, Section 4]. This algorithm works as follows. Starting from the bottom element, it iteratively applies Δ_λ to the current distance function generating the increasing chain $\mathbf{0} \sqsubseteq \Delta_\lambda(\mathbf{0}) \sqsubseteq \Delta_\lambda^2(\mathbf{0}) \sqsubseteq \cdots \sqsubseteq \Delta_\lambda^{k-1}(\mathbf{0}) \sqsubseteq \Delta_\lambda^k(\mathbf{0})$.

For each input instance, the comparison involves the following steps:

1. We run Algorithm 1, storing execution time, the number of solved transportation problems, and the number of coupling structures generated during the execution (i.e., the number of times a λ-discrepancy has been computed);
2. Then, on the same instance, we execute the value iteration algorithm until the running time exceeds that of step 1. We report the execution time, the number of solved transportation problems, and the number of iterations.
3. Finally, we report the error $\max_{s,t \in S} |\mathbf{d}_\lambda(s,t) - d(s,t)|$ between the distance $\mathbf{d}_\lambda$ computed in step 1 and the approximate result d obtained in step 2.

[3] `https://bitbucket.org/discoveri/probabilistic-bisimilarity-distances-probabilistic-automata`

Table 1 Comparison between Simple Policy and Value Iteration Algorithm. Average performance conducted on 100 randomly generated automata with number of states $n = 10..50$, nondeterministic out-degree $k = 1..3$, and probabilistic out-degree $p = 2..3$. Discount $\lambda = 0.8$; accuracy 0.000001.

$n = \|S\|$	Simple Policy Iteration			Value Iteration			Error
	time (sec)	# TP	# $\mathcal{C}$	time (sec)	# TP	# Iter	
10	0.254	356.3	23.2	0.291	733.3	4.3	0.06005
11	0.383	454	29	0.426	987	4.8	0.04971
12	0.549	564.5	35.3	0.613	1226.5	5	0.05156
13	0.742	678.5	42.6	0.820	1480	5.5	0.05252
14	1.099	809.6	50	1.212	1877.5	5.8	0.04841
15	1.668	957.8	58.6	1.879	2160.8	6	0.05431
20	4.405	1883.3	111	6.892	4661	7.8	0.03198
30	42.392	4570.2	251.2	44.263	14625.4	11	0.02026
40	160.725	8569.5	474.5	169.457	30713.6	14.4	0.00459
50	473.598	13877	748.6	490.575	56155.2	16,8	0.01224

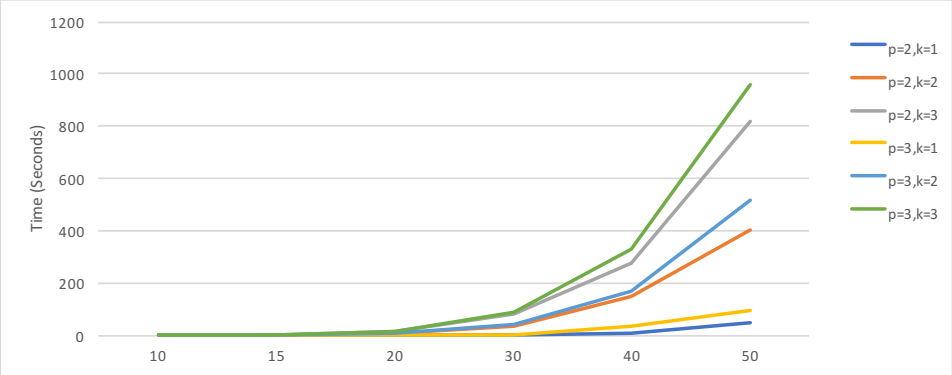


Figure 2 Average performance for the Simple Policy Iteration Algorithm conducted on 100 randomly generated automata varying number of states $n = 10..50$, nondeterministic out-degree $k = 1..3$, and probabilistic out-degree $p = 2..3$. Discount factor $\lambda = 0.8$; accuracy 0.000001.

This has been done for a collection of automata varying from 10 to 50 states. For each $n = 10, \ldots, 50$, we considered 100 randomly generated probabilistic automata, varying probabilistic out-degree and nondeterministic out-degree. Table 1 reports the average results of the comparison. Our algorithm is able to compute the solution before value iteration can under-approximate it with an error ranging from 0.004 to 0.06 which is a non negligible error considering that we fixed $\lambda = 0.8$ and the distance has values in $[0, 1]$.

Furthermore in Figure 2 we observe that the execution time of the simple policy iteration algorithm is particularly influenced by the degree of nondeterminism of the automaton. This may be explained by the fact that the current implementation uses a linear program for computing the λ-discrepancy (cf. Remark 11) which has $O(n^2k^2)$ variables and $O(n^2k^2)$ constraints where n and k are the number of states and the nondeterministic out-degree of the automaton, respectively.

Algorithm 2 extends the simple policy iteration algorithms proposed in [1, 37] for Markov chains. As pointed out in [36], implementations based on the decision procedure for the existential fragment of the first-order theory of the reals fail to handle Markov chains with a handful of states. For probabilistic automata, the algorithms in [6, 7] suffer from the same

Table 2 Average performance of Algorithm 2 conducted on 100 randomly generated automata with number of states $n = 10..50$, nondeterministic out-degree $k = 1..3$, and probabilistic out-degree $p = 2..3$. Discount $\lambda = 1$; accuracy 0.000001.

$n = \|S\|$	time (sec)	# TP	# $\mathcal{C}$	# outer-loops
10	0.310	392.6	23.8	1
11	0.460	509	30.3	1
12	0.665	651.5	37.2	1
13	0.921	807	44.5	1
14	1.205	952.6	51.5	1
15	1.657	1158.2	59.8	1
20	7.633	2278.5	108.8	1
30	29.430	4880.2	207	1
40	121.855	8245.3	340.3	1
50	381.563	12938	532.5	1

problem. The performance of Algorithm 2 is comparable to that of Algorithm 1 (cf. Table 2). Despite the fact that the simple policy algorithm is not guaranteed to be sound when the discount factor equals one, our experiments show that in practice a single iteration of the outer-loop of Algorithm 2 is often sufficient to yield the correct solution.

7 Conclusion and Future Work

We presented a novel characterization of the probabilistic bisimilarity distance of Deng et al. [12] as the solution of a simple stochastic game. Starting from it, we designed algorithms for computing the distances based on Condon's simple policy iteration algorithm. The correctness of Condon's approach relies on the assumption that the input game is stopping. This may not be the case for our probabilistic bisimilarity games when the discount factor is one. We overcame this problem by means of an improved termination condition based on the notion of self-closed relation due to Fu [17].

As in [37], our simple policy iteration algorithm has exponential worst-case time complexity. Nevertheless, experiments show that our method can compete in practice with the value iteration algorithm by Fu [17] which has theoretical polynomial-time complexity for $\lambda < 1$. To the best of our knowledge, our algorithm is the first practical solution for computing the bisimilarity distance when $\lambda = 1$, performing orders of magnitude faster than the existing solutions based on the existential fragment of the first-order theory of the reals [6, 7, 9].

As future work, we plan to improve upon the current implementation in the line of [38], by exploiting the fact that bisimilar states and probabilistic distance one [39] can be efficiently pre-computed before to start the policy iteration. We believe that this would yield a significant cut down in the time required to compute the discrepancy at each iteration which turned out to be the bottleneck of our algorithms.

More efficient algorithms might lead to the speedup of verification tools for concurrent probabilistic systems, as behavioral distances relate to the satisfiability of logical properties. For the case of Markov chains, in [8, 2] the variational difference between two states with respect to their probability of satisfying linear-time properties (eg., LTL formulas) is shown to be bound by the (undiscounted) probabilistic bisimilarity distance. Some preliminary results show that a similar bound holds also for probabilistic automata, though with additional subtleties that arise by the need of resolving the non-determinism.

We also plan to extend the work on approximated minimization [3, 4] to the case of probabilistic automata and explore the possible relation between the probabilistic bisimilarity distance with more expressive logics for concurrent probabilistic systems [6, 7, 27].

References

1 Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. On-the-Fly Exact Computation of Bisimilarity Distances. In *19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2013*, volume 7795 of *LNCS*, pages 1–15, 2013. doi:10.1007/978-3-642-36742-7_1.

2 Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. Converging from Branching to Linear Metrics on Markov Chains. In *12th International Colloquium Theoretical Aspects of Computing, ICTAC 2015*, volume 9399 of *Lecture Notes in Computer Science*, pages 349–367. Springer, 2015. doi:10.1007/978-3-319-25150-9_21.

3 Giovanni Bacci, Giorgio Bacci, Kim G. Larsen, and Radu Mardare. On the Metric-Based Approximate Minimization of Markov Chains. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017*, volume 80 of *LIPIcs*, pages 104:1–104:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.104.

4 Giovanni Bacci, Giorgio Bacci, Kim G. Larsen, and Radu Mardare. On the metric-based approximate minimization of Markov Chains. *J. Log. Algebr. Meth. Program.*, 100:36–56, 2018.

5 Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. MIT Press, 2008.

6 Krishnendu Chatterjee, Luca de Alfaro, Rupak Majumdar, and Vishwanath Raman. Algorithms for Game Metrics. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2008*, volume 2 of *LIPIcs*, pages 107–118. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2008. doi:10.4230/LIPIcs.FSTTCS.2008.1745.

7 Krishnendu Chatterjee, Luca de Alfaro, Rupak Majumdar, and Vishwanath Raman. Algorithms for Game Metrics (Full Version). *Logical Methods in Computer Science*, 6(3), 2010. doi:10.2168/LMCS-6(3:13)2010.

8 Di Chen, Franck van Breugel, and James Worrell. On the Complexity of Computing Probabilistic Bisimilarity. In *15th International Conference on Foundations of Software Science and Computational Structures, FOSSACS 2012*, volume 7213 of *Lecture Notes in Computer Science*, pages 437–451. Springer, 2012. doi:10.1007/978-3-642-28729-9_29.

9 Taolue Chen, Tingting Han, and Jian Lu. On Behavioral Metric for Probabilistic Systems: Definition and Approximation Algorithm. In *4th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2007*, pages 21–25. IEEE Computer Society, 2007. doi:10.1109/FSKD.2007.426.

10 Anne Condon. On Algorithms for Simple Stochastic Games. In *Advances In Computational Complexity Theory*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 51–72. DIMACS/AMS, 1990.

11 Anne Condon. The Complexity of Stochastic Games. *Inf. Comput.*, 96(2):203–224, 1992. doi:10.1016/0890-5401(92)90048-K.

12 Yuxin Deng, Tom Chothia, Catuscia Palamidessi, and Jun Pang. Metrics for Action-labelled Quantitative Transition Systems. *Electr. Notes Theor. Comput. Sci.*, 153(2):79–96, 2006. doi:10.1016/j.entcs.2005.10.033.

13 Josee Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for Labelled Markov Processes. *Theor. Comput. Sci.*, 318(3):323–354, 2004. doi:10.1016/j.tcs.2003.09.013.

14 Josée Desharnais, François Laviolette, and Mathieu Tracol. Approximate Analysis of Probabilistic Processes: Logic, Simulation and Games. In *5th International Conference on the Quantitative Evaluaiton of Systems, QEST 2008*, pages 264–273. IEEE Computer Society, 2008. doi:10.1109/QEST.2008.42.

15 Nathanaël Fijalkow, Bartek Klin, and Prakash Panangaden. Expressiveness of Probabilistic Modal Logics, Revisited. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017*, volume 80 of *LIPIcs*, pages 105:1–105:12. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.105.

16 Hongfei Fu. Computing Game Metrics on Markov Decision Processes. In *39th International Colloquium on Automata, Languages, and Programming, ICALP 2012*, volume 7392 of *Lecture Notes in Computer Science*, pages 227–238. Springer, 2012. doi:10.1007/978-3-642-31585-5_23.

17 Hongfei Fu. personal communication, 2013.

18 Alessandro Giacalone, Chi-Chang Jou, and Scott A. Smolka. Algebraic Reasoning for Probabilistic Concurrent Systems. In *Proceedings of the IFIP WG 2.2/2.3 Working Conference on Programming Concepts and Methods*, pages 443–458. North-Holland, 1990.

19 Felix Hausdorff. *Grundzüge der Mengenlehre*. Verlag Von Veit & Comp, Leipzig, 1914.

20 Bengt Jonsson and Kim G. Larsen. Specification and Refinement of Probabilistic Processes. In *6th Annual Symposium on Logic in Computer Science, LICS 1991*, pages 266–277. IEEE Computer Society, 1991. doi:10.1109/LICS.1991.151651.

21 Brendan Juba. On the Hardness of Simple Stochastic Games. Master's thesis, Carnegie Mellon University, Pittsburgh, PA, USA, May 2005.

22 Leonid Vitalevich Kantorovich. On the transfer of masses (in Russian). *Doklady Akademii Nauk*, 5(5-6):1–4, 1942. Translated in Management Science, 1958.

23 Peter Kleinschmidt and Heinz Schannath. A Strongly Polynomial Algorithm for the Transportation Problem. *Math. Program.*, 68:1–13, 1995. doi:10.1007/BF01585755.

24 Barbara König and Christina Mika-Michalski. (Metric) Bisimulation Games and Real-Valued Modal Logics for Coalgebras. In *29th International Conference on Concurrency Theory, CONCUR 2018*, volume 118 of *LIPIcs*, pages 37:1–37:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.CONCUR.2018.37.

25 Thomas Liggett and Steven A. Lippman. Stochastic Games with Perfect Information and Time Average Payoff. *SIAM Review*, 11(4):604–607, 1969. doi:10.1137/1011093.

26 Facundo Mémoli. Gromov-Wasserstein Distances and the Metric Approach to Object Matching. *Foundations of Computational Mathematics*, 11(4):417–487, 2011. doi:10.1007/s10208-011-9093-5.

27 Matteo Mio. On the Equivalence of Game and Denotational Semantics for the Probabilistic μ-calculus. *Logical Methods in Computer Science*, 8(2), 2012. doi:10.2168/LMCS-8(2:7)2012.

28 James B. Orlin. *On the Simplex Algorithm for Networks and Generalized Networks*, pages 166–178. Springer Berlin Heidelberg, 1985. doi:10.1007/BFb0121050.

29 Gabriel Peyré and Marco Cuturi. Computational Optimal Transport. *Foundations and Trends in Machine Learning*, 11(5-6):355–607, 2019. doi:10.1561/2200000073.

30 Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

31 Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley-Interscience series in Discrete Mathematics and Optimization. Wiley, 1999.

32 Roberto Segala. *Modeling and Verification of Randomized Distributed Real-time Systems*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1995.

33 Roberto Segala and Nancy A. Lynch. Probabilistic Simulations for Probabilistic Processes. In *5th International Conference on Concurrency Theory, CONCUR 1994*, volume 836 of *Lecture Notes in Computer Science*, pages 481–496. Springer, 1994. doi:10.1007/978-3-540-48654-1_35.

34 Lloyd S. Shapley. Stochastic Games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953. doi:10.1073/pnas.39.10.1095.

35 James K. Strayer. *Linear Programming and its Applications*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, NY, USA, 1989. doi:10.1007/978-1-4612-1009-2.

36 Qiyi Tang. *Computing Probabilistic Bisimilarity Distances*. PhD thesis, York University, Toronto, Canada, August 2018.

37 Qiyi Tang and Franck van Breugel. Computing Probabilistic Bisimilarity Distances via Policy Iteration. In *27th International Conference on Concurrency Theory, CONCUR 2016*, volume 59 of *LIPIcs*, pages 22:1–22:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.22.

38 Qiyi Tang and Franck van Breugel. Deciding Probabilistic Bisimilarity Distance One for Labelled Markov Chains. In *30th International Conference on Computer Aided Verification, CAV 2018*, volume 10981 of *Lecture Notes in Computer Science*, pages 681–699. Springer, 2018. doi:10.1007/978-3-319-96145-3_39.

39 Qiyi Tang and Franck van Breugel. Deciding Probabilistic Bisimilarity Distance One for Probabilistic Automata. In Sven Schewe and Lijun Zhang, editors, *29th International Conference on Concurrency Theory, CONCUR 2018*, volume 118 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:17. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.CONCUR.2018.9.

40 Franck van Breugel and James Worrell. The Complexity of Computing a Bisimilarity Pseudometric on Probabilistic Automata. In *Horizons of the Mind. A Tribute to Prakash Panangaden —Essays Dedicated to Prakash Panangaden on the Occasion of His 60th Birthday*, volume 8464 of *Lecture Notes in Computer Science*, pages 191–213. Springer, 2014. doi:10.1007/978-3-319-06880-0_10.

41 Cédric Villani. *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer, 2008.